



การเฝ้าระวังเพื่อการรู้จำภาษาธรรมชาติอย่างชาญฉลาด
Artificial Intelligence Application

รายงานฉบับสมบูรณ์

เสนอต่อ

ศูนย์เทคโนโลยีอิเล็กทรอนิกส์และคอมพิวเตอร์แห่งชาติ
สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติ
กระทรวงวิทยาศาสตร์และเทคโนโลยี

ได้รับทุนอุดหนุนโครงการวิจัย พัฒนาและวิศวกรรม
โครงการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทย ครั้งที่ 18
ประจำปีงบประมาณ 2558

โดย

นายภัคพล พงษ์ทวี

อาจารย์ ดร.ทัศนวรรณ ศูนย์กลาง

คณะวิทยาศาสตร์ มหาวิทยาลัยศิลปากร

กิตติกรรมประกาศ

โครงการการเยื้องเพื่อการรู้จำภาษาธรรมชาติอย่างชาญฉลาด ได้สำเร็จลุล่วงไปได้ด้วยดี ผู้พัฒนาขอขอบคุณ โครงการการแข่งขันพัฒนาโปรแกรมคอมพิวเตอร์แห่งประเทศไทย ครั้งที่ 18 จากศูนย์เทคโนโลยีอิเล็กทรอนิกส์คอมพิวเตอร์แห่งชาติ สำนักงานพัฒนาวิทยาศาสตร์และเทคโนโลยีแห่งชาติที่ให้ทุนสนับสนุนในการทำโครงการ ขอขอบคุณภาควิชาคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยศิลปากรที่จัดอบรม การเขียนโปรแกรมคอมพิวเตอร์ และขอขอบคุณ อาจารย์ ดร.ทัศนวรรณ ศูนย์กลาง ที่ให้คำแนะนำการพัฒนาโครงการในครั้งนี้

ชื่อโครงการ	การเยื้องเพื่อการรู้จำภาษาธรรมชาติอย่างชาญฉลาด
ผู้พัฒนา	นายภักพล พงษ์ทวี คณะวิทยาศาสตร์ มหาวิทยาลัยศิลปากร
อาจารย์ที่ปรึกษา	อาจารย์ ดร.ทัศนวรรณ ศูนย์กลาง

บทคัดย่อ

ระบบสนทนาโต้ตอบอัตโนมัติคือโปรแกรมคอมพิวเตอร์ที่ดำเนินการสนทนาโต้ตอบกับผู้ใช้ ซึ่งในปัจจุบันเราสามารถพัฒนาระบบสนทนาโต้ตอบอัตโนมัติได้ด้วยภาษา AIML แต่ว่าภาษา AIML นั้นมีไวยากรณ์เหมือนภาษา XML คือการใช้แท็กเปิดและปิดซึ่งทำให้ผู้สร้างสามารถอ่านโค้ดได้ยาก และโค้ดที่เขียนขึ้นมามีความยาวมาก จึงทำให้การพัฒนาเป็นไปได้อย่างล่าช้า อีกทั้งตัวแปลภาษาของ AIML ส่วนใหญ่ไม่รองรับการใช้นิพจน์กับภาษาไทยเนื่องจากภาษาอังกฤษจะมีการแบ่งคำด้วยช่องว่างซึ่งไม่พบในภาษาไทย ผู้พัฒนาจึงมีความคิดที่จะใช้ การเยื้องของโค้ดให้เป็นประโยชน์สำหรับการระบุรูปแบบของประโยคคำถามและคำตอบในการสร้างระบบสนทนาโต้ตอบอัตโนมัติ ซึ่งจะช่วยให้ผู้ที่กำลังศึกษาการสร้างระบบสนทนาโต้ตอบอัตโนมัติสามารถใช้งานได้ง่ายและยังทำให้สามารถพัฒนาระบบได้อย่างรวดเร็ว โครงการนี้จึงได้พัฒนาภาษาไอรินซึ่งเป็นภาษาที่ใช้การเยื้องเพื่อการรู้จำภาษาธรรมชาติอย่างชาญฉลาด พร้อมทั้งยังสร้างตัวแปลภาษาไอรินขึ้นมาจากการทดสอบโดยใช้กรณีตัวอย่าง 250 กรณีพบว่าตัวแปลภาษาไอรินนั้นสามารถทำงานได้อย่างถูกต้อง

คำสำคัญ: ระบบสนทนาโต้ตอบอัตโนมัติ, AIML, การเยื้อง

Project title	Indent to Recognize for Intelligent Natural language
Candidate	Pakkapon Phongthawee Faculty of Science Silpakorn University
Project Advisor	Dr.Tassanawan Soonklang

Abstract

Chatter bot is a program that performs interactive dialogue with users. Nowadays, we develop chatter bot by using AIML. AIML use syntax like XML by using opening tag and closing tag that make code difficult to read and the source code tend to be very long. Thus, the chatter bot is developed quite slow. Moreover, most of AIML's interpreters are not support using expression with Thai, since there is space between each word in English but not exist in Thai. So the candidate has an idea to use the indent of code to make the format of questions and answers which are used to create chatter bot. It's designed for learners who are studying to create chatter bot and it also help the process of developing chatterbot to be fast. This project creates IRIN, Indent to Recognize for Intelligent Natural Language, and its interpreter. The experiment results with 250 test cases show that IRIN interpreter can work correctly

Keyword: Chatter bot, AIML, Indent

สารบัญ

	หน้า
1. บทคัดย่อ	
1.1 บทคัดย่อภาษาไทย	ข
1.2 บทคัดย่อภาษาอังกฤษ	ค
2. สารบัญ	ง
3. บทนำ	1
4. วัตถุประสงค์และเป้าหมาย	1
5. รายละเอียดของการพัฒนา	
5.1 เนื้อเรื่องย่อ	2
5.2 ทฤษฎีหลักการและเทคนิคที่เกี่ยวข้อง	4
5.3 เครื่องมือที่ใช้พัฒนา	4
5.4 รายละเอียดของโปรแกรมที่ได้รับการพัฒนาเชิงเทคนิค	5
5.5 ขอบเขตและข้อจำกัดของโปรแกรม	5
6. กลุ่มผู้ใช้โปรแกรม	5
7. ผลการทดสอบโปรแกรม	5
8. ปัญหาและอุปสรรค	9
9. แนวทางในการพัฒนาและประยุกต์ร่วมกับงานอื่น ในขั้นต่อไป	9
10. ข้อเสนอแนะ	10
11. เอกสารอ้างอิง	10
12. สถานที่ติดต่อของผู้พัฒนาและอาจารย์ที่ปรึกษา	10
13. ภาคผนวก	11

บทนำ

ระบบสนทนาโต้ตอบอัตโนมัติคือโปรแกรมคอมพิวเตอร์ที่ดำเนินการสนทนาโต้ตอบกับผู้ใช้โดยโปรแกรมดังกล่าวได้รับการออกแบบมักจะมีส่วนร่วมในการพูดคุยเล็กๆ เพื่อความสนุกสนานและนอกจากนี้ระบบสนทนาโต้ตอบอัตโนมัติยังถูกออกแบบมาเพื่อใช้งานให้เกิดประโยชน์ เช่น งานบริการลูกค้า, เลขาส่วนตัว เป็นต้น ซึ่งในปัจจุบันมีการความสนใจในการพัฒนาระบบสนทนาโต้ตอบอัตโนมัติเป็นอย่างมากโดยเฉพาะระบบที่รองรับภาษาไทย เช่น ซิมซิมิ (simsimi.com), ฟาไฮ (fahsai.in.th), ด.ช.หน้าหมา (dogdiri.com) เป็นต้น

ภาษา AIML พัฒนาโดย Dr. Richard S. Wallace เมื่อปี พ.ศ.2544 เป็นภาษาที่ใช้ในการสร้างระบบสนทนาโต้ตอบอัตโนมัติโดยมีตัวแปลภาษา (Interpreter) ในหลายภาษา เช่น Python, C++, C#, Java, Lisp, Pascal, Python, Ruby เป็นต้น ซึ่งภาษา AIML มีไวยากรณ์เหมือนภาษา XML คือการใช้แท็กเปิดและปิดซึ่งทำให้ผู้สร้างสามารถอ่านโค้ดได้ยาก และโค้ดที่เขียนขึ้นมีความยาวมาก จึงทำให้การพัฒนาเป็นไปได้อย่างล่าช้า อีกทั้งตัวแปลภาษาของ AIML ส่วนใหญ่ไม่รองรับการใช้งานกับภาษาไทยเนื่องจากภาษาอังกฤษจะมีการแบ่งคำด้วยช่องว่างซึ่งไม่พบในภาษาไทย

ดังนั้นผู้พัฒนาจึงมีความคิดที่ศึกษาการเยื้องเพื่อการรู้จำภาษาธรรมชาติอย่างชาญฉลาด (Indent to Recognize for Intelligent Natural language : IRIN) หรือเรียกอีกชื่อว่า ภาษาไอริน โดยใช้การเยื้องของโค้ดให้เป็นประโยชน์ในการบอกรูปแบบของประโยคคำถามและคำตอบในการสร้างระบบสนทนาโต้ตอบอัตโนมัติซึ่งจะทำให้โค้ดอ่านง่ายและสั้นลงอย่างมาก ตัวอย่าง เช่น ระบบสนทนาโต้ตอบอัตโนมัติ ELIZA เมื่อเขียนด้วยภาษา AIML โค้ดจะยาวถึง 1,226 บรรทัด (<https://goo.gl/8RwfDi>) แต่เมื่อเขียนด้วยภาษาไอริน โค้ดจะสั้นลงเหลือเพียง 301 บรรทัดเท่านั้น (<https://goo.gl/jUv8to>) ซึ่งเป็นมิตรต่อผู้ที่กำลังศึกษาการสร้างระบบสนทนาโต้ตอบอัตโนมัติและยังทำให้สามารถพัฒนาระบบได้อย่างรวดเร็ว อีกทั้งยังพัฒนาตัวแปลภาษาให้สามารถใช้งานกับภาษาไทยและภาษาอังกฤษได้เป็นอย่างดี

วัตถุประสงค์และเป้าหมายของโครงการ

1. เพื่อสร้างภาษาที่สามารถใช้ในการสร้างระบบสนทนาโต้ตอบอัตโนมัติได้ง่ายขึ้น
2. เพื่อสร้างตัวแปลภาษาของภาษาไอรินที่สร้างขึ้นจากภาษา Coffeescript ซึ่งตัวแปลภาษาที่สร้างขึ้นสามารถใช้งานกับภาษาไทยและภาษาอังกฤษได้เป็นอย่างดี

รายละเอียดการพัฒนา

เนื้อเรื่องย่อ

ภาษาไอรินเป็นภาษาที่ใช้การเยื้องของโค้ดให้เป็นประโยชน์ โดยใช้สำหรับการบอกว่าประโยคดังกล่าวเป็นคำถามหรือคำตอบของระบบโต้ตอบอัตโนมัติ โดยการพิมพ์ไม่เยื้องจะหมายถึงคำถามและการเยื้องเข้าไป 1 ชั้นจะหมายถึงคำตอบ และหากมีการเยื้องเข้าไปอีกจะเป็นคำถามสลับกันไป

1	กินข้าวหรือยัง	1	คำถาม
2	กินข้าวแล้วค่ะ	2	คำตอบ
3	อืมแล้วค่ะ	3	คำตอบ
4	ข้าวอร่อยไหม	4	คำถาม
5	อร่อยมากเลยคะ	5	คำตอบ
6	เยี่ยมยอดเลยคะ	6	คำตอบ
7	กินข้าวกับอะไร	7	คำถาม
8	กินกับผัดกระเพราคะ	8	คำตอบ
9	ยังเลยคะ	9	คำตอบ
10	ยังหิวอยู่เลยคะ	10	คำตอบ
11	หาอะไรกินกัน	11	คำถาม
12	ได้เลยคะ	12	คำตอบ
13	*	13	คำถาม
14	ขอภัยคะ ดิฉันไม่เข้าใจคำถาม	14	คำตอบ

ภาพที่ 1 ตัวอย่างโค้ดภาษาไอริน

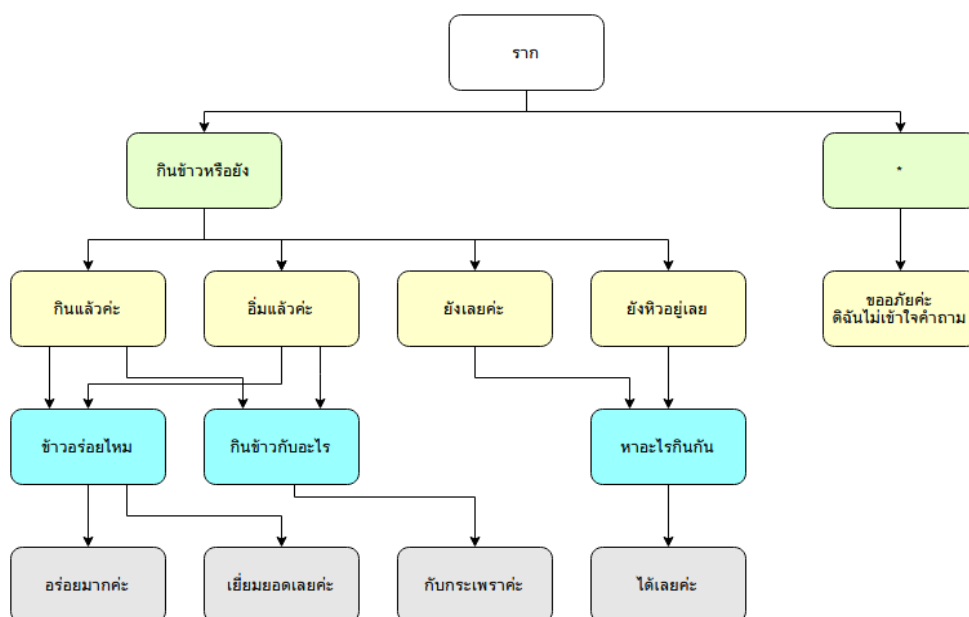
จากภาพที่ 1 จะพบ จะสามารถอธิบายรูปแบบคำถามและคำตอบได้ดังโค้ดด้านขวา นั่นคือคำถามกินข้าวหรือยัง โดยมีคำตอบ 4 รูปแบบได้แก่ กินข้าวแล้วค่ะ,อืมแล้วค่ะ,ยังเลยคะ และยังหิวอยู่เลยคะ ซึ่งหากระบบโต้ตอบอัตโนมัติเลือกที่จะตอบว่า อืมแล้วค่ะ ก็จะมีคำถามที่เยื้องเข้าไปอีก 2 คำถามคือ ข้าวอร่อยไหมและกินข้าวกับอะไร

นอกจากนี้ภาษาไอรินยังมีความสามารถอื่นอีก เช่น การใช้งานนิพจน์,หัวข้อเรื่อง,เงื่อนไขในบรรทัด,การกำหนดค่าตัวแปร เป็นต้น ซึ่งสามารถอ่านเพิ่มเติมได้ในภาคผนวก

สำหรับตัวแปลภาษาที่สร้างขึ้นจะเขียนด้วยภาษา คอฟฟี่สคริป (CoffeeScript) เนื่องจากภาษาคอฟฟี่สคริป เป็นภาษาที่เขียนสั้นและง่าย อีกทั้งยังสามารถแปลงเป็นภาษาจาวาสคริป (JavaScript) ได้ ซึ่งภาษาจาวาสคริปรองรับการแบบข้ามระบบปฏิบัติการ (Cross-platform) ซึ่งทำให้ตัวแปลภาษาที่เขียนขึ้นสามารถประยุกต์ใช้กับโปรแกรมอื่นที่เขียนขึ้นใหม่ได้ง่าย รองรับการใช้งานทั้งผ่านเบราว์เซอร์และผ่านโหนดเจส (Node.JS)

ซึ่งตัวแปลภาษาไอรินนั้นเป็นคลาส (Class) ที่ใช้งานโดยการส่งค่าพารามิเตอร์ที่อยู่ไฟล์และฟังก์ชันเรียกกลับ (Callback function) เข้าไปยังคอนสตรัคเตอร์ (Constructor) โดยคอนสตรัค

เตอร์นี้จะทำการแปลโค้ดภาษาไอรินที่เป็นการเยื้องให้อยู่ในรูปแบบโครงสร้างข้อมูลต้นไม้เพื่ออำนวยความสะดวกค้นหา



ภาพที่ 2 โครงสร้างต้นไม้ที่เกิดจากการแปลโค้ดในภาพที่ 1

โดยเมื่อทำการแปลภาษาเสร็จแล้วจะทำการเรียกใช้งานฟังก์ชันเรียกกลับ และเมื่อต้องการให้ระบบโต้ตอบอัตโนมัติตอบคำถาม ให้นำเข้าข้อมูลผ่านการส่งค่าผ่านพารามิเตอร์ไปยังเมธอด reply ของคลาส แล้วเมธอดนี้จะทำการค้นหาคำตอบในต้นไม้ที่คอนสตรัคเตอร์ได้สร้างขึ้น ตัวแปลภาษาจะทำการค้นหาโหนดลูกของโหนดราก ซึ่งบรรจุประโยชน์การสนทนา เมื่อนำไปเปรียบเทียบกับทำการแปลงนิพจน์ในภาษาไอรินเป็นนิพจน์ปกติ (Regular Expression) โดยทางผู้พัฒนาได้เขียนวิธีการแปลงนิพจน์ให้รองรับกับการใช้งานทั้งภาษาไทยและภาษาอังกฤษ ซึ่งหากการพบโหนดที่เข้าเงื่อนไขให้ตั้งโหนดที่เข้าเงื่อนไขเป็นโหนดแม่และสุ่มดึงประโยคจากโหนดลูกมาตอบ เมื่อตอบแล้ว ให้กำหนดโหนดลูกที่นำมาตอบเป็นโหนดแม่ เมื่อมีการนำเข้าข้อมูลครั้งถัดไปให้ทำการเปรียบเทียบโดยใช้นิพจน์ปกติ กับโหนดลูกของโหนดแม่ หากเข้าเงื่อนไขก็ทำซ้ำดังข้อความด้านบน แต่หากไม่พบโหนดลูกให้ทำการเปรียบเทียบโดยใช้นิพจน์ปกติ กับโหนดลูกของโหนดรากทั้งหมดและทำซ้ำ

ตัวภาษาที่ทำการเขียนขึ้นด้วยภาษาคอปปีสคริปนั้นจะได้รับการทดสอบการทำงานด้วย กัป (Gulp) ด้วยชุดทดสอบทั้งสิ้น 250 ชุด ทำซ้ำชุดละ 10 ครั้งเพื่อให้มั่นใจว่าตัวแปลภาษาที่สร้างขึ้นนั้นสามารถทำงานได้ถูกต้อง โดยสามารถดาวน์โหลดชุดทดสอบได้ที่ <https://goo.gl/cVfKiu>

ทฤษฎีหลักการและเทคนิคที่เกี่ยวข้อง

1. ต้นไม้ (Tree)

ต้นไม้คือโครงสร้างแบบไม่เป็นเส้นตรงประกอบด้วยโหนดหรือจุดยอดและขอบโดยโหนดหมายถึงสิ่งที่สามารถกำหนดชื่อและเก็บข้อมูลได้ ขอบคือเส้นที่เชื่อมกันระหว่างสองโหนด การกำหนดเส้นทางจากโหนดหนึ่งไปอีกโหนดหนึ่งโดยไม่ซ้ำกันบนต้นไม้เรียกว่าพาทบนโครงสร้างต้นไม้มีโหนดพิเศษเรียกว่ารากซึ่งเป็นโหนดที่มีพาทไปหาตัวมันจากโหนดใดๆในต้นไม้ได้เพียงพาทเดียวเท่านั้น และการเก็บข้อมูลของภาษาไอรินถูกออกแบบมาโดยใช้ต้นไม้เพื่อให้สะดวกต่อการค้นหาคำตอบ

2. นิพจน์ปกติ (Regular expression)

นิพจน์ปกติคือรูปแบบของอักขระ ที่ใช้เพื่อค้นหาข้อความในสายอักขระ เนื่องจากภาษาไอรินรองรับการใช้นิพจน์ดังนั้นการเปรียบเทียบนิพจน์จึงใช้การแปลงนิพจน์ในภาษาไอรินเป็นนิพจน์ปกติเพื่อใช้เปรียบเทียบในภาษา Coffeescript

3. Shunting-yard algorithm

Shunting-yard algorithm ใช้สำหรับการแปลงนิพจน์ทางคณิตศาสตร์ให้อยู่ในรูปแบบของ Reverse Polish notation เพื่อให้สามารถคำนวณนิพจน์คณิตศาสตร์สามารถทำได้ถูกต้องตามลำดับความสำคัญของเครื่องหมาย ซึ่งในภาษาไอรินนั้นจะต้องใช้การคำนวณนิพจน์คณิตศาสตร์ในการกำหนดตัวแปร และการใช้งานเงื่อนไขในบรรทัด

เครื่องมือที่ใช้พัฒนา

Software

Atom.io 1.0.0

Coffeescript 1.10.0

Gulp 3.9.0

Node.js 4.2.2 LTS

Ubuntu 15.04

Hardware

Asus A45VM (RAM: 8GB, CPU: Intel i7 2.2-3.2GHZ, HDD: 750GB)

รายละเอียดโปรแกรมที่จะพัฒนา

Input/Output Specification

Input: โค้ดภาษาไอริน นำเข้าผ่านทางคอนสตรัคเตอร์ของคลาส และคำถามที่
รับเข้าผ่านทางเมธอด reply

Output: คำตอบของคำถามที่ได้รับผ่านทางเมธอด reply

Functional Specification

ตัวแปลภาษาสามารถทำงานได้ถูกต้องตามรูปแบบภาษาที่ได้กำหนดไว้

ขอบเขตและข้อจำกัดของโปรแกรมที่พัฒนา

ตัวแปลภาษาที่พัฒนาขึ้นได้รับการทดสอบเพื่อใช้งานภาษาไทยและภาษาอังกฤษเท่านั้นอาจ
เกิดปัญหาขึ้นเมื่อนำไปใช้งานกับภาษาอื่น

กลุ่มผู้ใช้โปรแกรม

ภาษาไอรินที่พัฒนาขึ้นเพื่อช่วยให้ผู้ที่เริ่มศึกษาการพัฒนาระบบโต้ตอบอัตโนมัติสามารถสร้างระบบ
โต้ตอบอัตโนมัติได้ง่าย และเป็นทางและเป็นทางเลือกของผู้ที่พัฒนาระบบโต้ตอบอัตโนมัติอยู่แล้วให้มีทางเลือก
การสร้างระบบได้ง่ายขึ้น

ผลของการทดสอบโปรแกรม

การเขียนเพื่อการรู้จำภาษาธรรมชาติอย่างชาญฉลาดหรือภาษาไอริน แบ่งการพัฒนาเป็น 2 ส่วนคือ
การพัฒนาภาษาและตัวแปลภาษา โดยรูปแบบการเขียนของภาษาไอรินสามารถเปรียบเทียบกับภาษา AIML
ได้ผลดังตาราง

ตารางที่ 1 ตารางเปรียบเทียบคำสั่งระหว่างภาษา AIML และภาษาไอริน

AIML	ไอริน	ความหมาย
<aiml version="XXX">	ไม่จำเป็นต้องบอกรุ่นของภาษาที่ใช้ในการเขียน	บอกรุ่นของภาษาที่ใช้เขียน
<topic name="XXX">	-> XXX	กำหนดหัวข้อเรื่องในการ สนทนา
<category>	ไม่จำเป็นต้องใช้ในภาษาไอริน	ใช้สำหรับครอบคลุมของ คำถามและคำตอบไว้ด้วยกัน ซึ่งไม่จำเป็นในภาษาไอริน

AIML	ไอริน	ความหมาย
<pattern>HELLO BOT</pattern>	hello bot	ใช้สำหรับกำหนดประโยคคำถาม
<template> Hello human! </template>	Hello human!	ใช้สำหรับกำหนดประโยค
<that>XXX</that>	XXX yyy zzzz	ใช้สำหรับอ้างอิงการตอบครั้งล่าสุด ในภาษาไอรินใช้การเยื้องเพื่อสร้างการสนทนาต่อเนื่อง
<star index="N"/>	{N}	ใช้สำหรับแสดงข้อความในตำแหน่งของดอกจันที่ N เมื่อ N เป็นจำนวนนับ
<that index="N">	สามารถใช้การกำหนดค่าตัวแปรแทนได้เช่น That's ok {prev<-That's ok} แล้วเรียกใช้ผ่านตัวแปรชื่อ prev	ใช้สำหรับแสดงประโยคการตอบออกมาประโยคที่ N ก่อนหน้า
<input index="N">	ใช้ {0} สำหรับแสดงคำถามของผู้ใช้ล่าสุดจากนั้นให้กำหนด {prev<-{0}} เพื่อเรียกใช้ค่าผ่านตัวแปร prev ในภายหลัง	สำหรับแสดงประโยคคำถามของผู้ใช้ ประโยคที่ N ก่อนหน้า
<thatstar index="N">	สามารถใช้วิธี {prev<-{N}} แล้วเรียกใช้ค่าผ่านตัวแปร prev ในภายหลังได้	สำหรับแสดงบริเวณของ * ที่บริเวณ N ก่อนหน้า
<topicstar index="N">	ไม่รองรับในภาษาไอริน	ใช้สำหรับบอกหัวข้อเรื่องที่คุยกัน N หัวข้อก่อนหน้า
<get name="XXX" />	{XXX}	ใช้สำหรับเข้าถึงค่าตัวแปร
<set name="XXX">YYY</set>	{XXX<-YYY}	ใช้สำหรับกำหนดค่าตัวแปร
<bot name="XXX" />	{XXX}	ใช้สำหรับเข้าถึงค่าตัวแปร
<date />	ไม่รองรับในภาษาไอริน	เรียกใช้เพื่อแสดงเวลา
<id />	ไม่รองรับในภาษาไอริน	บอกรหัสสำหรับระบุตัวตน

AIML	ไอริน	ความหมาย
<size />	ไม่รองรับในภาษาไอริน	บอกจำนวนของคำถามที่มีอยู่
<version />	ไม่รองรับในภาษาไอริน	บอกรุ่นของภาษาที่ใช้เขียน
<uppercase>XXX</uppercase>	ในภาษาไอรินไม่จำเป็นต้องใช้การจัดรูปแบบตัวอักษร เนื่องจากใช้วิธีการตรวจสอบ regular expression โดยใช้ flag i นั่นคือไม่สนใจรูปแบบการ พิมพ์ ดังนั้น Hello กับ hello ถือว่าเหมือนกันในภาษาไอริน	ใช้สำหรับจัดรูปแบบให้ข้อความเป็นตัวพิมพ์ใหญ่
<lowercase>XXX</lowercase>		ใช้สำหรับจัดรูปแบบให้ข้อความเป็นตัวพิมพ์เล็ก
<formal>XXX</formal>		ใช้สำหรับจัดรูปแบบให้ศัพท์เฉพาะเป็นตัวพิมพ์ใหญ่ในตัวแรก
<sentence>XXX</sentence>		ใช้สำหรับการจัดรูปแบบข้อความให้เป็นตัวแรกพิมพ์ใหญ่ พร้อมปิดท้ายด้วยจุด
<condition name="X" value="Y"> <condition name="X"> <conditon>	{{x}}=={{y}}?:}	ใช้สำหรับตรวจสอบเงื่อนไขว่ารูปคำ X เท่ากับ คำ Y หรือไม่ โดยภาษาไอรินสามารถสามารถตรวจสอบได้ว่า เท่ากับ,ไม่เท่ากับ,มากกว่า,น้อยกว่า,มากกว่าเท่ากับ,น้อยกว่าเท่ากับ
<if expr="X<Y"> </if>	{{x}}<{{y}}?:}	ใช้สำหรับเปรียบเทียบค่า x กับ y ซึ่งสามารถใช้เครื่องหมายต่างๆได้
<random>	Hello Nice to see you Hi	ใช้สำหรับสุ่มคำตอบของประโยค โดยภาษาไอรินจะใช้วิธีการให้ลำดับการเยื้องอยู่ลำดับเดียวกันเพื่อสุ่ม
	ไม่จำเป็นต้องใช้ในภาษาไอริน	ใช้สำหรับบอกว่าเป็นข้อมูลแบบรายการ

AIML	ไอริน	ความหมาย
<gossip>	ภาษาไอริน ใช้วิธีการเก็บในหน่วยความจำ จึงไม่จำเป็นต้องเขียนลงไฟล์	ใช้สำหรับเก็บองค์ความรู้ใหม่ในไฟล์
<srai>XXX</srai> <sr/>	XXX Hello Good Morning Nice to see you	ใช้สำหรับดึงคำตอบของคำถามอื่นมาในการตอบ ซึ่งภาษาไอรินสามารถเขียนอยู่ในรูปแบบ หลายคำถาม แล้วจึงเอียงมาเป็นคำตอบแทนได้
<person>XXX</person>	ไม่รองรับในภาษาไอริน	เปลี่ยนสรรพนามบุรุษที่ 1 เป็นสรรพนามบุรุษที่ 2 พร้อมทั้งเปลี่ยนสรรพนามบุรุษที่ 2 เป็นบุรุษที่ 1
<person2>XXX</person2>	ไม่รองรับในภาษาไอริน	เปลี่ยนสรรพนามบุรุษที่ 1 เป็นบุรุษที่ 3 พร้อมทั้งเปลี่ยนสรรพนามบุรุษที่ 3 เป็นบุรุษที่ 1
<gender>XXX</gender> ตัวอย่าง <gender>She told him to take a hike.</gender>	สามารถใช้เงื่อนไขแทนได้ ตัวอย่าง {{gender}}==male?he:she} told {{gender}}==male?her:him} to take a hike.	ใช้สำหรับแปลงคำสรรพนามให้ตรงกับเพศของผู้ใช้ ดังตัวอย่าง หากผู้ใช้เป็นหญิงก็จะถูกแปลงเป็น He told her to take a hike. ซึ่งไม่รองรับในภาษาไอริน แต่สามารถใช้แทนกันได้โดยการใช้เงื่อนไขแทน
<think>	{}	ใช้สำหรับให้ระบบไม่ตอบกลับไม่รองรับภาษาไอริน แต่ภาษาไอรินสามารถตอบกลับเป็น “ ” ได้โดยเขียนว่า {}
<learn filename="X" />	-> X.irin	ใช้สำหรับนำเข้าคำสั่งจากไฟล์อื่น

AIML	ไอริน	ความหมาย
<system>	ไม่มีคำสั่งนี้ในภาษาไอริน เนื่องจากอาจทำให้เกิดช่องโหว่ด้านความปลอดภัยได้	เรียกใช้คำสั่งระดับโอเอส
<javascript>,<perl>,etc.		เรียกใช้คำสั่งภาษาอื่นในโค้ดตัวเอง

สำหรับตัวแปลภาษาไอรินสามารถตรวจสอบความถูกต้องได้ของการทำงานโดยใช้กัปทดสอบโดยมีชุดทดสอบทั้งสิ้นจำนวน 250 ชุด ทำซ้ำชุดละ 10 ครั้งเพื่อให้มั่นใจว่าตัวแปลภาษาที่สร้างขึ้นนั้นสามารถทำงานได้ถูกต้อง โดยสามารถดาวน์โหลดชุดทดสอบได้ที่ <https://goo.gl/cVfKiu>

ปัญหาและอุปสรรค

ตัวแปลภาษาไอรินใช้การตรวจสอบโดยใช้ชุดทดสอบ ซึ่งอาจทำให้ไม่ครอบคลุมข้อผิดพลาดที่อาจเกิดขึ้นได้

แนวทางในการพัฒนาและประยุกต์ใช้ร่วมกับงานอื่นๆ ในขั้นต่อไป

สำหรับตัวแปลภาษาไอรินเป็นโครงการเปิดโค้ด ท่านสามารถดาวน์โหลดโค้ดของภาษาไอรินเพื่อพัฒนาต่อได้ที่ <https://github.com/pureexe/irin-lang> และท่านยังสามารถลองใช้งานภาษาไอรินโดยไม่จำเป็นต้องติดตั้งผ่านทางออนไลน์ได้ที่ <https://ภาษา.ไอริน.ไทย/ลองใช้>

สำหรับการประยุกต์ใช้กับงานอื่น สามารถติดตั้งเพื่อใช้งานได้ผ่านทาง npm โดยการใช้คำสั่ง `npm install irin-lang` และยังสามารถติดตั้งผ่าน bower ได้โดยการใช้คำสั่ง `bower install irin-lang` และหากท่านไม่ได้ใช้งานทั้ง npm และ bower ท่านสามารถดาวน์โหลดโค้ดที่ได้ทำการแปลงเป็นจาวาสคริปและลดขนาดแล้วได้ที่ <https://github.com/pureexe/irin-lang/releases> โดยตัวแปลภาษาไอรินนั้นรองรับทั้งการใช้งานผ่าน Node.js และ CommonJS โดยการใช้คำสั่ง `var Irin = require("irin-lang");` หรือใช้ผ่านเบราว์เซอร์โดยการใช้คำสั่ง `<script src="path/to/irin-lang.min.js"></script>` และเริ่มการแปลภาษาโดยการใช้คำสั่ง `var bot = new Irin("path/to/main.irin",function(){});` เมื่อแปลภาษาเสร็จแล้วตัวแปลภาษาจะทำการเรียกฟังก์ชันเรียกกลับ ท่านสามารถถามคำถามกับระบบโต้ตอบอัตโนมัติได้โดยการใช้คำสั่ง `bot.reply("คำถาม");`

ข้อสรุปและข้อเสนอแนะ

จากการทดสอบด้วยชุดทดสอบผ่านกัป พบว่าตัวแปลภาษาไอรินสามารถทำงานได้ถูกต้อง

เอกสารอ้างอิง

- Alice Bot. **AIML reference**. [ออนไลน์] เข้าถึงเมื่อ 8 ตุลาคม 2558. เข้าถึงทาง
<http://www.alicebot.org/documentation/aiml-reference.html>
- Mozilla. **Regular Expression**. [ออนไลน์] เข้าถึงเมื่อ 8 ตุลาคม 2558. เข้าถึงได้ทาง
https://developer.mozilla.org/th/docs/Web/JavaScript/Guide/Regular_Expressions
- Wikipedia. **AIML**. [ออนไลน์] เข้าถึงเมื่อ 8 ตุลาคม 2558. เข้าถึงได้ทาง
<https://en.wikipedia.org/wiki/AIML>
- Wikipedia. **Chatterbot**. [ออนไลน์] เข้าถึงเมื่อ 8 ตุลาคม 2558. เข้าถึงได้ทาง
<https://en.wikipedia.org/wiki/Chatterbot>
- Wikipedia. **ELIZA**. [ออนไลน์] เข้าถึงเมื่อ 8 ตุลาคม 2558. เข้าถึงได้ทาง
<https://en.wikipedia.org/wiki/ELIZA>
- Wikipedia. **Shunting-yard algorithm**. [ออนไลน์] เข้าถึงเมื่อ 8 ตุลาคม 2558. เข้าถึงได้ทาง
https://en.wikipedia.org/wiki/Shunting-yard_algorithm
- บุญเจริญ ศิริเนาวกุล,รศ.ดร. และพิพัฒน์ ศุภศิริสันต์,ผศ. (2550). **โครงสร้างข้อมูลและอัลกอริทึม**.
 กรุงเทพฯ: สำนักพิมพ์ท้อป

สถานที่ติดต่อของผู้พัฒนาและอาจารย์ที่ปรึกษา

นายภาคพล พงษ์ทวี

สถานที่ติดต่อ: 70/111 หมู่ 5 ตำบลอ่างทอง อำเภอเมือง จังหวัดราชบุรี 70000

โทรศัพท์มือถือ: 080-5797336

อีเมล: phongthawee_p@silpakorn.edu

อาจารย์ ดร.ทัศนวรรณ ศูนย์กลาง

สถานที่ติดต่อ: วิทย์1 ชั้น6 เลขที่6 ถนนราชมรรคาใน ตำบลพระปฐมเจดีย์ อำเภอเมืองนครปฐม

จังหวัดนครปฐม 73000

โทรศัพท์มือถือ: 081-8556062

อีเมล: tasanawa@su.ac.th

ภาคผนวก

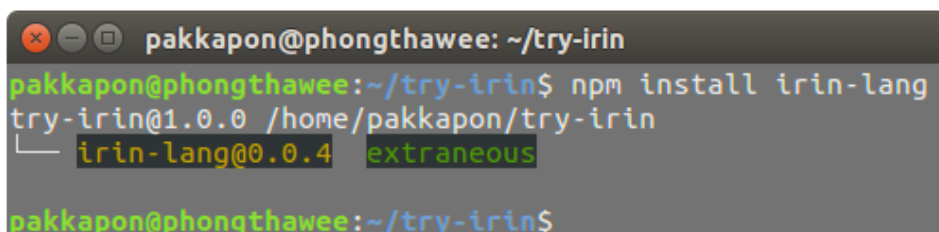
คู่มือการติดตั้งอย่างละเอียด

การดาวน์โหลด

ท่านสามารถติดตั้งตัวแปลภาษาไอรินเพื่อใช้กับโปรแกรมของท่านได้โดยการติดตั้งได้ผ่านทาง npm, bower หรือสามารถติดตั้งผ่านการดาวน์โหลดไฟล์ที่ถูกแปลงเป็นจาวาสคริปและลดขนาดแล้วได้

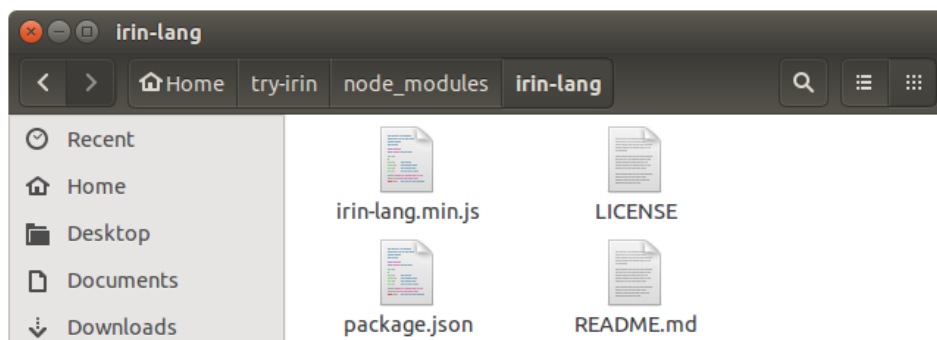
การติดตั้งผ่าน npm

หากท่านใช้งาน npm ท่านสามารถติดตั้งตัวแปลภาษาไอรินได้โดยการใช้งานคำสั่ง
npm install irin-lang



```
pakkapon@phongthawee: ~/try-irin
pakkapon@phongthawee:~/try-irin$ npm install irin-lang
try-irin@1.0.0 /home/pakkapon/try-irin
└─┬ irin-lang@0.0.4 extraneous
pakkapon@phongthawee:~/try-irin$
```

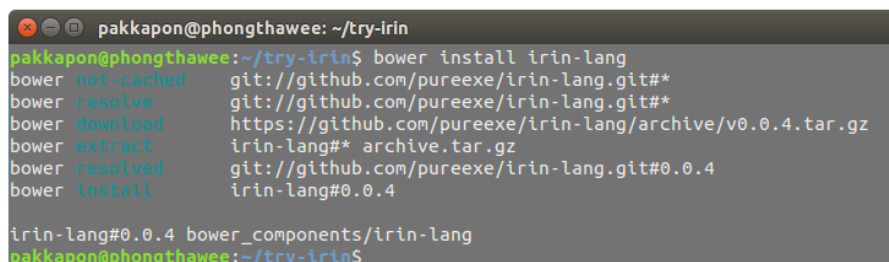
เมื่อการติดตั้งเสร็จสมบูรณ์จะปรากฏข้อความดังภาพ



จะปรากฏไฟล์ขึ้นมาที่ โฟลเดอร์/node_modules/irin-lang

การติดตั้งผ่าน bower

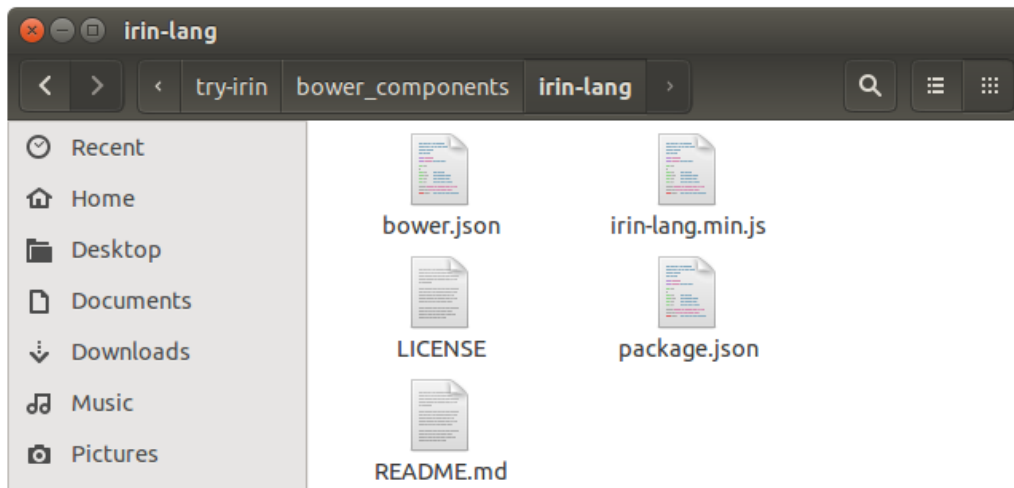
หากท่านใช้งาน bower ท่านสามารถติดตั้งตัวแปลภาษาไอรินได้โดยการใช้งานคำสั่ง
bower install irin-lang



```
pakkapon@phongthawee: ~/try-irin
pakkapon@phongthawee:~/try-irin$ bower install irin-lang
bower not-cached git://github.com/pureexe/irin-lang.git#*
bower resolve git://github.com/pureexe/irin-lang.git#*
bower download https://github.com/pureexe/irin-lang/archive/v0.0.4.tar.gz
bower extract irin-lang# archive.tar.gz
bower resolved git://github.com/pureexe/irin-lang.git#0.0.4
bower install irin-lang#0.0.4

irin-lang#0.0.4 bower_components/irin-lang
pakkapon@phongthawee:~/try-irin$
```

เมื่อการติดตั้งเสร็จสมบูรณ์จะปรากฏข้อความดังภาพ



จะปรากฏไฟล์ขึ้นที่ โฟลเดอร์/bower_components/irin-lang

การติดตั้งโดยการดาวน์โหลดไฟล์

ท่านสามารถดาวน์โหลดไฟล์ได้ที่ <https://github.com/pureexe/irin-lang/releases>

Latest release

v0.0.4

40fa5d9

Release: v0.0.4

pureexe released this 8 minutes ago

- fix: `->` on difference directory
- fix: expression `-` and `/` is wrong because stack popping error.
- fix: regular expression escape
- fix: declare variable which isn't following top-down law
- fix: `###` multiline comment working incorrectly `###`
- add: support `data<-(other_variable)+5` in header
- add: error Unexpected declaration in header.
- add: error Variable name must not start with number.
- add: error unexpected indentation
- remove: error Header must have no indent. (replace by unexpected indentation)

Downloads

irin-lang.min.js	11.4 KB
Source code (zip)	
Source code (tar.gz)	

โดยให้ท่านดาวน์โหลดไฟล์ชื่อ `irin-lang.min.js` เพื่อนำไปใช้งาน

การติดตั้ง

ใช้งานผ่าน Node.JS หรือ CommonJS

ในการใช้งานผ่าน Node.JS หรือ CommonJS สามารถนำเข้าตัวแปลภาษาได้ผ่านทางคำสั่ง require โดยสามารถเขียนคำสั่งได้ดังนี้

```
var Irin = require("irin-lang")
```

ใช้งานผ่านเบราว์เซอร์

ในการใช้งานผ่านเบราว์เซอร์ สามารถนำเข้าตัวแปลภาษาได้ผ่านทางแท็ก script โดยสามารถเขียนคำสั่งได้ดังนี้

```
<script src="path/to/irin-lang.min.js"></script>
```

การใช้งานในโปรแกรม

การแปลภาษา

ก่อนที่จะเริ่มโต้ตอบกับระบบสนทนาได้ตอบอัตโนมัติที่เขียนขึ้นจากภาษาไอรินได้ จำเป็นต้องทำการแปลภาษาก่อน ซึ่งสามารถทำการแปลภาษาได้โดยการเขียนโค้ดภาษาจาวาสคริปดังนี้

```
var bot = new Irin("ที่อยู่ไฟล์.irin",function(err){
    if(err){
        //เกิดข้อผิดพลาดขึ้นขณะแปลภาษา
    }else{
        //การแปลภาษาเสร็จสมบูรณ์
    }
});
```

การโต้ตอบ

การโต้ตอบกับโค้ดภาษาไอรินที่ถูกแปลภาษาเรียบร้อยแล้วนั้นทำได้โดยการเขียนโค้ดภาษาจาวาสคริปดังนี้

```
bot.reply("คำถาม")
```

การใช้งานภาษาไอริน

การเยื้องเพื่อการรู้จำภาษาธรรมชาติอย่างชาญฉลาด (Indent to Recognize for Intelligent Natural language : IRIN) หรือเรียกอีกชื่อว่า ภาษาไอริน เป็นภาษาที่แบบมาให้ใช้การเยื้องในการแทนตำแหน่งของคำถามและคำตอบในการสร้างระบบสนทนาโต้ตอบอัตโนมัติซึ่งจะทำให้โค้ดอ่านง่ายและสั้นลงอย่างมาก ซึ่งมีไวยากรณ์ในการเขียนดังนี้

การเยื้อง

ในภาษาไอรินใช้การเยื้องในการแทนตำแหน่งของคำถามและคำตอบโดยบรรทัดที่ไม่มีการเยื้องเลย จะทำหน้าที่เป็นคำถาม และบรรทัดที่เยื้องเข้าไปจะเป็นคำตอบ เมื่อเยื้องเข้าไปอีกจะเป็นคำถาม สลับกันไปเรื่อยๆ ตัวอย่างเช่น

```
1 | สวัสดี
2 |   มีอะไรให้ช่วยหรือคะ
```

เมื่อถามระบบว่า สวัสดี ระบบจะตอบว่า มีอะไรให้ช่วยหรือคะ ซึ่งจะเห็นได้ว่า สวัสดี ทำหน้าที่เป็นคำถาม และ มีอะไรให้ช่วยหรือคะ เป็นคำตอบ

อีกทั้งภาษาไอรินยังใช้การเยื้องในการแยกคำถามแต่ละข้อออกจากกัน ตัวอย่างเช่น

```
1 | สวัสดี
2 |   มีอะไรให้ช่วยหรือคะ
3 |   ลาก่อน
4 |   ลาก่อนคะ
```

จะเห็นว่าโค้ดดังกล่าวมี 2 คำถามคือ สวัสดี กับลาก่อน

แต่หากไม่มีการเยื้องในบรรทัดที่ติดกันจะถือว่าเป็นคำถามที่มีคำตอบร่วมกัน ตัวอย่างเช่น

```
1 | สวัสดี
2 |   ดีจ้า
3 |   หวัดดี
4 |   มีอะไรให้ช่วยหรือคะ
```

จะสังเกตว่าบรรทัดที่ 1,2 และ 3 ไม่มีการเยื้อง จึงถือว่าเป็นคำถามร่วมกัน ดังนั้นเมื่อถามระบบว่า สวัสดี, ดีจ้า, หรือหวัดดี ระบบจะตอบกลับมาเหมือนกันคือ มีอะไรให้ช่วยหรือคะ

และหากไม่มีการเยื้องในบรรทัดที่ติดกันของคำตอบจะถือว่าเป็นคำตอบร่วมกันของคำถาม โดยคำตอบร่วมกัน จะใช้การสุ่มหนึ่งในคำตอบออกมาตอบ ตัวอย่างเช่น

```
1 | สวัสดี
2 |   สวัสดีค่ะ
3 |   ยินดีต้อนรับค่ะ
4 |   มีอะไรให้ช่วยหรือคะ
```

หากเราถามระบบว่าสวัสดี ระบบอาจตอบว่า สวัสดีค่ะ, ยินดีต้อนรับค่ะ หรือมีอะไรให้ช่วยหรือคะ

ทำนองเดียวกันเราสามารถใช้คำถามร่วมและคำตอบร่วมพร้อมกันได้ ตัวอย่างเช่น

1	สวัสดี
2	ดีจ้า
3	หวัดดี
4	สวัสดีค่ะ
5	ยินดีต้อนรับค่ะ
6	มีอะไรให้ช่วยหรือคะ

เมื่อถามระบบว่าสวัสดี,ดีจ้า หรือหวัดดี ระบบอาจตอบว่า สวัสดีค่ะ, ยินดีต้อนรับค่ะ หรือมีอะไรให้ช่วยหรือคะ

อีกทั้งเรายังสามารถใช้การเยื้องเพื่อเป็นการบอกการสนทนาต่อเนื่องได้ โดยการเพิ่มการเยื้องเข้าไปต่อจากคำตอบของคำถามก่อนหน้า ตัวอย่างเช่น

1	สวัสดี
2	สวัสดีค่ะ
3	นอนละ
4	ฝันดีค่ะ
5	หิวข้าว
6	ไปกินสิคะ
7	ไม่มีเงิน
8	เดี๋ยวให้ยืมคะ
9	ไม่คืนได้ไหม
10	ไม่ได้ค่ะ
11	ไม่หิวแล้ว
12	แล้วจะบอกว่าหิวข้าวทำไมคะ

สังเกตว่าบริเวณคำว่าหิวข้าวมีการเยื้องถัดเข้าไปต่อจากคำตอบของคำถามแรก นั้นหมายถึงการสนทนาต่อเนื่อง เมื่อถามระบบว่า สวัสดี ตัวแปลภาษาจะตอบว่า สวัสดีค่ะ แต่เมื่อเราถามระบบต่อด้วยคำว่าหิวข้าว ระบบจะตอบกลับมว่า ไปกินสิคะ ซึ่งช่วงนี้จะเป็นการสนทนาต่อเนื่องซึ่งตอนนี้มี 2 ทางเลือกคือ ไม่มีเงิน กับไม่หิวแล้ว ซึ่งหากถามระบบต่อว่าไม่มีเงิน ระบบจะตอบว่า เดี๋ยวให้ยืมคะ แต่หากเรากรอกว่านอนละ ระบบจะออกจากการสนทนาต่อเนื่องแล้วตอบว่า ฝันดีค่ะ

ความคิดเห็น

ความคิดเห็นหรือคอมเมนต์ (Comment) ใช้เพื่อบอกความคิดเห็นของเราลงไปในโค้ด โดยความคิดเห็นนั้นมีไว้เพื่อให้มนุษย์อ่านเท่านั้น เพื่อให้ผู้เขียนโค้ดสามารถจำได้ว่าโค้ดที่ตัวเองเขียนมีไว้ใช้งานอย่างไร โดยความคิดเห็นนั้นจะไม่ถูกนำไปประมวลผลโดยตัวแปลภาษา

ความคิดเห็นบรรทัดเดียว

ใช้เครื่องหมาย # เพื่อระบุว่าเป็นความคิดเห็นบรรทัดเดียว โดยตัวแปลภาษาจะไม่นำข้อความหลังเครื่องหมายไปประมวลผล ตัวอย่างเช่น

```
1  #นี่คือความคิดเห็น
2  สวัสดี
3  มีอะไรให้ช่วยหรือคะ
```

ความคิดเห็นหลายบรรทัด

ใช้เครื่องหมาย ### เพื่อระบุว่าเป็นความคิดเห็นหลายบรรทัด โดยต้องปิดท้ายความเห็นด้วย ### เสมอ โดยตัวแปลภาษาจะไม่นำข้อความที่อยู่ในช่วงของเครื่องหมายดังกล่าวไปประมวลผล ตัวอย่างเช่น

```
1  ###
2  สิ่งที่ปรากฏส่วนนี้
3  จะไม่ทำงาน
4  ###
5  สวัสดี
6  มีอะไรให้ช่วยหรือคะ
```

นิพจน์

นิพจน์ (Expression) คือ รูปแบบสำหรับเปรียบเทียบข้อความว่าเข้าเงื่อนไขหรือไม่ หากนิพจน์ตรงกับรูปแบบของศัพท์ที่กรอกเข้ามาจะทำการเรียกคำตอบของคำถามนั้นถัดไป โดยภาษาไอรินมีนิพจน์ต่างๆดังต่อไปนี้

ดอกจันท์

ดอกจันท์ * คือ ตรงกับทุกตัวอักษร ไม่ว่าตัวอักษรใดก็ตามจะเป็นจริงเสมอ ตัวอย่างเช่น

```
1  สวัสดี
2  สวัสดีค่ะ
3  *
4  ขอภัยค่ะ ดิฉันไม่เข้าใจ
```

เมื่อเราถามคำถามใดกับระบบก็ตามที่ไม่ใช่ สวัสดี ระบบจะตอบกลับว่า ขอภัยค่ะ ดิฉันไม่เข้าใจ เนื่องจากนิพจน์ดอกจันท์แทนตัวอักษรใดก็ได้

นอกจากนี้เรายังใช้นิพจน์ดอกจันท์แทนบางส่วนของประโยคได้เช่น

```
1  ผมว่าเธอ*มาก
2  ทำไมคุณถึงคิดอย่างนั้นละ
```

เมื่อถามระบบว่า ผมว่าเธอน่ารักมาก กับถามระบบว่า ผมว่าเธอสวยมาก ระบบจะตอบกลับมาว่า ทำไมคุณถึงคิดอย่างนั้นละ

แต่ว่าดอกจันทน์นั้นไม่ถือว่าตรงกับคำว่าว่างเปล่า ต้องมีอย่างน้อย 1 ตัวอักษรขึ้นไป ตัวอย่างเช่นโค้ดด้านบน หากถามระบบว่า ผมว่าเธอมาก จะสังเกตได้ว่า บริเวณที่เป็นดอกจันทน์จะไม่ตรงกับตัวอักษรใดๆเลย ดังนั้นระบบจะไม่ตอบว่าทำไมคุณถึงคิดอย่างนั้นละ จึงขอให้ระวังในส่วนนี้อาไว้ด้วย

วงเล็บ

() คือ เลือก คำใดคำหนึ่งจากในวงเล็บ แบ่งคำออกจากกันด้วยเครื่องหมาย | ตัวอย่างเช่น

1		หิว(ใหม่ หรือไม่)
2		หิวแล้วค่ะ

เมื่อถามระบบว่า หิวใหม่ หรือถามว่า หิวหรือไม่ ระบบจะตอบกลับมาว่า หิวแล้วแล้วค่ะ เนื่องจากคำว่าใหม่และคำว่าหรือไม่เป็นคำใดคำหนึ่งจากในวงเล็บ แต่หากถามว่าหิวมะ ระบบจะไม่ตอบว่าหิวแล้วค่ะ เนื่องจากคำว่ามะ ไม่ได้ตรงกับคำใดคำหนึ่งในวงเล็บ

วงเล็บก้ามปู

[] คือ อาจ มีคำใดคำหนึ่งจากในวงเล็บ แบ่งคำออกจากกันด้วยเครื่องหมาย | ตัวอย่างเช่น

1		สวัสดี[ครับ ค่ะ]
2		สวัสดีค่ะ

เมื่อถามระบบว่า สวัสดี,สวัสดีครับ หรือสวัสดีค่ะ ระบบจะตอบกลับมาว่า สวัสดีค่ะ แต่หากถามระบบว่าสวัสดีนะ ระบบจะไม่ตอบกลับมาว่าสวัสดีค่ะ เนื่องจากไม่ได้บอกกับระบบว่าจากมีคำว่านะ อยู่ด้านหลัง

และเรายังสามารถใช้เครื่องหมายวงเล็บก้ามปูร่วมกับดอกจันทน์ได้ ตัวอย่างเช่น

1		สวัสดี[*]
2		สวัสดีค่ะ

นิพจน์นี้มีความหมายว่าอาจมีตัวอักษรใดก็ได้ ดังนั้นจะตรงกับข้อความที่มี หรือไม่มีตัวอักษรต่อท้ายก็ได้ ดังนั้นจึงสามารถตรงกับข้อความว่า สวัสดีจ้า สวัสดีค่ะ หรือสวัสดีก็ได้

ส่วนหัว

ส่วนหัวมีไว้สำหรับการประกาศตัวแปรโดยส่วนหัวจะเริ่มต้นด้วยขีดกลาง - 3 ขีด และปิดด้วยขีดกลางอีก 3 ขีด โดยการกำหนดตัวแปรนั้นจะใช้เครื่องหมาย <- โดยการตั้งชื่อตัวแปรนั้น สามารถตั้งโดยใช้ตัวหรือตัวอักษรก็ได้แต่ห้ามขึ้นต้นด้วยตัวเลข ตัวอย่างเช่น

```

1  ---
2  name <- 'ไอริน'
3  gender <- 'หญิง'
4  age <- 18
5  ---

```

โดยจากตัวอย่างโค้ดคือการประกาศตัวแปร 3 ตัว ตัวแปรแรกชื่อ name เก็บค่า ไอริน ตัวแปรที่ 2 ชื่อ gender เก็บค่าหญิง ตัวแปรที่ 3 เก็บค่า 18

การเข้าถึงตัวแปร

การเข้าถึงตัวแปรสามารถทำได้โดยใช้เครื่องหมายปีกกา { } โดยแบ่งตัวแปรเป็น 2 ประเภทได้แก่ ตัวแปรทั่วไปและตัวแปรตามเหตุการณ์

ตัวแปรทั่วไป

คือตัวแปรที่ประกาศไว้บนส่วนหัวของไฟล์ สามารถเรียกใช้งานได้โดยเขียนว่า {ชื่อตัวแปร} แล้วตัวแปลภาษาจะทำการแทนค่าของตัวแปรไปยังบริเวณที่ {ชื่อตัวแปร} ปรากฏอยู่ ตัวอย่างเช่น

```

1  ---
2  name <- 'ไอริน'
3  ---
4  ชื่ออะไร
5  ดิฉันชื่อ{name}ค่ะ

```

เมื่อถามระบบว่า ชื่ออะไร ระบบจะตอบว่า ดิฉันชื่อไอรินค่ะ เนื่องจากตัวแปลภาษาได้แทนค่าของตัวแปร name ลงไปยังบริเวณที่เขียนว่า {name}

นอกจากนี้เรายังสามารถใช้ตัวแปรทั่วไปกับคำถามได้ด้วย ตัวอย่างเช่น

```

1  ---
2  name <- 'ไอริน'
3  ---
4  {name}
5  เรียกดิฉันทำไมคะ

```

เมื่อเราถามระบบว่า ไอริน ระบบจะตอบกลับว่า เรียกดิฉันทำไมคะ เนื่องจากค่าของตัวแปร name ซึ่งมีค่าเป็นไอริน จะถูกแทนที่ลงในคำถาม ดังนั้นคำถามนั้นจึงเป็นคำว่า ไอริน ทำให้เมื่อถามว่าไอรินแล้วระบบจึงตอบว่า เรียกดิฉันทำไมคะ

ตัวแปรตามเหตุการณ์

ตัวแปรตามเหตุการณ์จะเกิดขึ้นเมื่อมีการใช้นิพจน์ () และ * เนื่องจากนิพจน์ 2 ตัวนี้จะต้องมีค่าใดค่าหนึ่งแน่นอน โดยการเข้าถึงจะใช้ {ตำแหน่งของนิพจน์} โดยนับเป็นเลขเริ่มต้นจาก 1 ตัวอย่างเช่น

```
1  ผม(ชอบ|เกลียด)*
2  ทำไมคุณถึงได้{1}พวก{2}ละ
```

เมื่อถามระบบว่า ผมชอบแมว ระบบจะตอบกลับมาว่า ทำไมคุณถึงชอบพวกแมวละเนื่องจากคำว่าชอบ ถูกนำไปแทนค่าในตัวแปร {1} และคำว่า แมว ถูกนำไปแทนค่าในตัวแปร {2}

การเปลี่ยนค่าตัวแปร

เราสามารถเปลี่ยนแปลงค่าของตัวแปรแบบทั่วไป ที่มีอยู่เดิมได้โดยการกำหนดค่าตัวแปรเข้าไปใหม่แบบพลวัตโดยทำการใช้เครื่องหมาย <- เพื่อกำหนดค่าตัวแปรเข้ามาใหม่ภายใต้เครื่องหมายปีกกา ตัวอย่างเช่น

```
1  ---
2  name <- ไอริน
3  isHungry <- กำลังหิวเลยล่ะ
4  ---
5  กินข้าวกัน
6  {name}{isHungry}{isHungry<-อิ่มแล้วล่ะ}
```

จากตัวอย่าง เมื่อถามระบบว่า กินข้าวกัน ระบบจะตอบกลับมาว่า ไอรินกำลังหิวเลยล่ะ เมื่อถามระบบอีกครั้งว่า กินข้าวกัน ระบบจะตอบกลับมาว่า ไอรินอิ่มแล้วล่ะ

โดยการกำหนดค่านั้นจะทำเมื่อเจอวงเล็บปีกกาที่มีเครื่องหมาย <- ดังนั้นหากเราสลับตำแหน่งของวงเล็บปีกกาในโค้ดตัวอย่างด้านบน {name}{isHungry<-อิ่มแล้วล่ะ}{isHungry} เมื่อถามว่ากินข้าวกัน ระบบจะตอบกลับมาว่า ไอรินอิ่มแล้วล่ะ เนื่องจากโค้ดดังกล่าวได้ทำการกำหนดค่าตัวแปรใหม่ก่อนนำไปแสดงผล

อีกทั้งภายใต้เครื่องหมายปีกกายังสามารถกำหนดค่าได้โดยการใช้เครื่องหมายปีกกา ยังสามารถใช้เครื่องหมาย บวก ลบ คูณ หาร ในการช่วยกำหนดค่าได้ ตัวอย่างเช่น

```
1  ---
2  sheep <- 0
3  ---
4  นับแกะ
5  {sheep<-{sheep}+1}แกะ{sheep}ตัว
```

หากถามระบบว่านับแกะ ระบบจะตอบว่า แกะ1ตัว หากถามระบบอีกครั้งว่า นับแกะ ระบบจะตอบว่าแกะ 2 ตัวและเพิ่มขึ้นเรื่อยๆตามลำดับ

สังเกตว่าคำสั่ง `{sheep<-{sheep}+1}` จำเป็นต้องใส่วงเล็บปีกกาที่ชื่อตัวแปรในฝั่งขวาของเครื่องหมาย `<-` เสมอ หากไม่ใส่เครื่องหมายปีกกา ระบบจะถือว่าคำว่า `sheep` เป็นอักขระ ไม่ใช่ตัวแปร

และเรายังสามารถนำค่าที่ได้มาจากตัวแปรตามเหตุการณ์กำหนดค่าใส่ในตัวแปรทั่วไปเพื่อใช้งานค่าในภายหลังได้ ดังตัวอย่าง

```

1  ---
2  user <- คุณ
3  ---
4  สวัสดี
5      สวัสดีค่ะ คุณชื่ออะไรหอรคะ
6      [ผม|ฉัน][ชื่อ]*[ครับ|ค่ะ]
7      สวัสดีค่ะคุณ{1}{user<-{1}}
8  หิวข้าว
9      ทำไม{user}ไม่ไปกินข้าวละ

```

เมื่อถามระบบว่า หิวข้าว ระบบจะตอบว่า ทำไมคุณไม่ไปกินข้าวละ เมื่อถามระบบต่อว่า สวัสดี ระบบจะตอบว่า สวัสดีค่ะ คุณชื่ออะไรหอรคะ จึงถามระบบต่อว่า ผมชื่อเพียวครับ ระบบตอบกลับมาว่า สวัสดีค่ะคุณเพียว และเมื่อถามระบบต่อว่า หิวข้าว ระบบจะตอบว่า ทำไมเพียวไม่ไปกินข้าวละ จะสังเกตว่าคำถามว่าหิวข้าวทั้ง 2 ครั้งชื่อผู้ใช้จะแตกต่างกัน

เงื่อนไขในบรรทัด

เราสามารถทำให้คำตอบการสนทนาเปลี่ยนไปได้โดยการเงื่อนไขในบรรทัดได้ โดยใช้เครื่องหมาย ? ตามด้วยคำตอบเมื่อเป็นจริง แบ่งด้วย : แล้วตามด้วยเงื่อนไขเมื่อเป็นเท็จ โดยเครื่องหมายในการเปรียบเทียบที่สามารถใช้ได้ ได้แก่ และ (&&), หรือ (||), นิเสธ (!), เท่ากับ (==), มากกว่าเท่ากับ (>=), น้อยกว่าเท่ากับ (<=), มากกว่า (>), น้อยกว่า (<), บวก(+), ลบ (-), คูณ (*) และหาร (/) ตัวอย่างเช่น

```

1  เธออายุเท่าไร
2      อายุ 18 ค่ะ แล้วคุณอายุของคุณเท่าไรคะ
3      [ผม|ฉัน][อายุ]*[ครับ|ค่ะ]
4      คุณนี่ดู{1}<18?เด็ก:แก่}จังเลยคะ

```

เมื่อถามระบบว่า เธออายุเท่าไร ระบบจะตอบว่า อายุ 18 ค่ะ แล้วคุณอายุของคุณเท่าไรคะ เมื่อถามกลับว่า อายุ 20 ครับ ระบบจะตอบกลับมาว่า คุณนี่ดูแก่จังเลยคะ แต่หากถามกลับว่า อายุ 12 ครับ ระบบจะตอบว่า คุณนี่ดูเด็กจังเลยคะ

หัวข้อเรื่อง

ในภาษาไอริน ฟังก์ชันนั้นจะใช้เครื่องหมาย -> แทนการกำหนดหัวข้อเรื่อง ทั้งการประกาศหัวข้อเรื่อง และการนำเข้าสู่หัวข้อเรื่อง โดยการประกาศและใช้หัวข้อเรื่องจะ -> ชื่อหัวข้อ แตกกันที่การประกาศหัวข้อเรื่อง นั้นนั้นก่อนหน้าเครื่องหมาย -> จะไม่มีการเยื้องเลย หากมีการเยื้องจะเป็นการนำเข้าสู่หัวข้อเรื่องนั้น ตัวอย่างเช่น

```
1 | สวัสดี
2 |   สวัสดีค่ะ
3 |   ->จากลา
4 | ->จากลา
5 |   ไปละ
6 |   ลาก่อนค่ะ
```

เมื่อถามระบบว่า สวัสดี ระบบจะตอบว่า สวัสดีค่ะ หลังจากนั้นถามว่า ไปละ ต่อระบบจะตอบว่า ลาก่อนค่ะ เนื่องจากเยื้องของคำว่าสวัสดีค่ะ มีการเรียกใช้หัวข้อจากลา ซึ่งจะเป็นการดึงเอาคำถามและคำตอบที่อยู่ในหัวข้อจากลาไปเชื่อมอยู่กับคำถามสวัสดี ดังนั้นคำถามที่อยู่ต่อจากคำว่าสวัสดีค่ะก็คือคำว่าไปละ หรือก็คือผลลัพธ์ที่ได้จะเหมือนกับการเขียนโค้ดดังนี้

```
1 | สวัสดี
2 |   สวัสดีค่ะ
3 |   ไปละ
4 |   ลาก่อนค่ะ
```

และยังสามารถเรียกหัวข้อจากไฟล์อื่นได้โดยการนำหัวข้อเรื่องใส่ไว้ในไฟล์อื่นแล้วตั้งชื่อไฟล์ด้วยนามสกุล .irin แล้วทำการเรียกใช้ด้วยคำสั่งหัวข้อเรื่อง

ตัวอย่างเช่น ไฟล์แรกเขียนว่า

```
1 | สวัสดี
2 |   สวัสดีค่ะ
3 |   ->goodbye.irin
```

ไฟล์ goodbye.irin เขียนว่าให้ทำการเขียนประโยคการโต้ตอบได้เลยโดยไม่ต้องใส่ชื่อหัวข้อเรื่อง

```
1 | ไปละ
2 |   ลาก่อนค่ะ
```

จะเห็นว่าเมื่อรันแล้วผลลัพธ์จะสามารถใช้งานได้เหมือนกับการเขียนหัวข้อเรื่องไว้ในไฟล์เดียวกัน

แต่หากการเขียนเครื่องหมาย -> โดยไม่มีการเยื้องและตามด้วย .irin นั้นถือเป็นการนำเข้าไฟล์มาใช้งานไม่ใช่การประกาศหัวข้อเรื่อง ตัวอย่างเช่น

```

1 | สวัสดี
2 |   สวัสดีค่ะ
3 | ->goodbye.irin

```

ผลลัพธ์ที่ได้จะทำงานเหมือนกับโค้ดดังต่อไปนี้

```

1 | สวัสดี
2 |   สวัสดีค่ะ
3 | ไปละ
4 |   ลาก่อนค่ะ

```

บนลงล่าง

สำหรับภาษาไอรินจะมีการเรียงลำดับความสัมพันธ์โดยการทำงานโค้ดจากบนลงล่าง หากโค้ดที่มีลำดับการเยื้องเดียวกันจะทำการเลือกโค้ดที่อยู่ด้านบนเสมอ ตัวอย่างเช่น

```

1 | สวัสดี
2 |   สวัสดีค่ะ
3 | สวัสดี
4 |   สวัสดีจ้า

```

เมื่อถามระบบว่าสวัสดี ระบบจะตอบว่าสวัสดีค่ะ เสมอเนื่องจากสวัสดีในบรรทัดที่หนึ่งเป็นบรรทัดแรก ที่ตรงกับเงื่อนไข ดังนั้นจึงตอบว่าสวัสดีค่ะเสมอ

นอกจากนี้การทำงานจากบนลงล่างยังมีผลต่อการกำหนดค่าตัวแปรอีกด้วยตัวอย่างเช่น

```

1 | ---
2 | name <- ไอริน
3 | name <- อลิส
4 | ---

```

จากโค้ดตัวอย่างมีการประกาศตัวแปรชื่อ name ซ้ำกัน 2 ครั้งดังนั้นแล้วค่าของตัวแปรจะเป็น ไอริน เนื่องจากโค้ดพบว่า

การกำหนดหัวข้อซ้ำกันจตัวแปรภาษาจะเลือกใช้งานหัวข้อที่เจอเป็นหัวข้อแรก ตัวอย่างเช่น

```

1 | หิวข้าวไหม
2 |   หิวแล้วค่ะ
3 |   ->ไปกินข้าว
4 |
5 | ->ไปกินข้าว
6 |   ไปกินข้าวกัน
7 |   ได้เลยค่ะ
8 |
9 | ->ไปกินข้าว
10 |   ไปกินข้าวกัน
11 |   ไววันหลังนะคะ

```

เมื่อถามระบบว่า หิวข้าวไหม ระบบตอบว่าหิวแล้วค่ะ เมื่อถามระบบต่อว่า ไปกินข้าวกัน ระบบจะตอบว่าได้เลยค่ะเสมอ เนื่องจากเจอหัวข้อเรื่องที่บรรทัดที่ 5 ก่อน

ประโยชน์ของการใช้งานโค้ดแบบบนลงล่างคือไม่สามารถเขียนทับตัวแปรที่ของไฟล์อื่นได้เหมาะสมกับกรณีไฟล์ภาษาไอรินมาจากการเขียนโดยคนละคนกันตัวอย่างเช่น

```
1  ---
2  name <- ไอริน
3  ---
4
5  -> default.irin
6  -> aichan/main.irin
7  -> alice/main.irin
8  -> arisa/main.irin
```

ถ้าทั้ง 4 ไฟล์เขียนโดยคนละคนกันแล้วใช้ ในแต่ละไฟล์มีการประกาศ ตัวแปรชื่อ name เหมือนกัน ตัวแปรทั้งหมดนั้นจะถูกเปลี่ยนค่าเป็น ไอรินทันทีเพื่อการอำนวยความสะดวกในการเขียนโค้ดของท่าน

นอกจากนี้แล้วท่านยังสามารถอ่านเอกสารวิธีการใช้งานภาษาไอรินและตัวแปลภาษาไอริน เพิ่มเติมได้ที่ <https://ภาษา.ไอริน.ไทย> และสามารถลองใช้งานภาษาไอรินออนไลน์ได้ที่ <https://ภาษา.ไอริน.ไทย/ลองใช้>